UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/709,522	05/11/2004	Krishna Mohan ITIKARLAPALLI	ORCL-003	3521

51121 7590 12/09/2010

LAW FIRM OF NAREN THAPPETA
C/o Landon-IP Inc.,
1725 Jamieson Avenue
Alexandria, VA 22314

EXAMINER	
SANDERS, AARON J	

ART UNIT	PAPER NUMBER
2168	

NOTIFICATION DATE	DELIVERY MODE
12/09/2010	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

lfnt2000@yahoo.com
oracle@iphorizons.com
intercomm@iphorizons.com

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/709,522
Filing Date: May 11, 2004
Appellant(s): ITIKARLAPALLI ET AL.

Narenda Reddy Thappeta
Registration No.: 41,416
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 21 September 2010 appealing from the Office action mailed 19 April 2010.

(1) Real Party in Interest

The examiner has no comment on the statement, or lack of statement, identifying by name the real party in interest in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The following is a list of claims that are rejected and pending in the application:

1-10, 13-21, and 25

(4) Status of Amendments After Final

The examiner has no comment on the appellant's statement of the status of amendments after final rejection contained in the brief.

(5) Summary of Claimed Subject Matter

The examiner has no comment on the summary of claimed subject matter contained in the brief.

(6) Grounds of Rejection to be Reviewed on Appeal

The examiner has no comment on the appellant's statement of the grounds of rejection to be reviewed on appeal. Every ground of rejection set forth in the Office action from which the appeal is taken (as modified by any advisory actions) is being maintained by the examiner except for the grounds of rejection (if any) listed under the subheading "WITHDRAWN REJECTIONS." New grounds of rejection (if any) are provided under the subheading "NEW GROUNDS OF REJECTION."

(7) Claims Appendix

The examiner has no comment on the copy of the appealed claims contained in the Appendix to the appellant's brief.

(8) Evidence Relied Upon

5,701,480	RAZ	12-1997
5,781,910	GOSTANIAN et al.	7-1998
5,857,204	LORDI et al.	1-1999

Applicant's Admitted Prior Art, Fig. 1 and Specification pars. 3-7 and 22-33

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1-2, 5-10, 13, 16-17, 20-21 and 25 are rejected under 35 U.S.C. 103(a) as being unpatentable over Applicant's Admitted Prior Art, Fig. 1 and Specification pars. 3-7 and 22-33 ("AAPA"), in view of Gostanian et al., U.S. 5,781,910 ("Gostanian"), and in view of Lordi et al., U.S. 5,857,204 ("Lordi").

Note on claim interpretation: The term "if" denotes an optionally recited limitation and optionally recited limitations are not guaranteed to take place and are therefore not required to be taught, see MPEP § 2106 Section II(C). Even so, the references teach the optional limitations.

1. AAPA teaches "*A method of implementing atomic transactions in a system, said method comprising*," see Fig. 1 and par. 23, "FIG. 1 contains pseudo-code illustrating the manner in which an example atomic transaction is implemented in a prior approach."

AAPA teaches "*specifying in said user program a plurality of combinations, wherein each of said plurality of combinations contains said transaction identifier, a task procedure, and a rollback procedure*," see Fig. 1, par. 23, "For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures… Account1( ) is shown containing program logic in lines 110 through 199,"

par. 24, "Line 110 is shown containing a call to task procedure P1( ). Line 115 is shown containing a call to task procedure P2( )," and par. 25, "Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )," where the claimed "combinations" are the referenced Account( ), P( ), and do-reverse-of-P( ) combinations and the claimed "transaction identifier" is the referenced "Account( )," since it is the name (i.e. "identifier") of the transaction. Also note that according to Applicant's specification at par. 69, "Even though the example above are shown specifying the combination in the form of a single line of code (procedure call), multiple lines can be used in alternative embodiments." Thus, it is irrelevant that the referenced "combinations" are not contained in a single procedure call.

AAPA teaches "*wherein each combination indicates that the rollback procedure is to be executed if the execution of the corresponding task procedure in the combination is completed and if said atomic transaction is to be aborted*," see Fig. 1 and pars. 25-26, "Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( ). The occurrence of an error generally represents an example situation in which the atomic transaction is to be aborted."

AAPA teaches "*wherein said task procedure implements a part of said atomic transaction and said rollback procedure is designed to rollback said task procedure*," see Fig. 1 and par. 23, "For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures."

AAPA teaches "*executing a set of task procedures in a sequential order according to said user program, wherein said set of task procedures are contained in said task procedures*

specified in said plurality of combinations," see Fig. 1 and par. 24, "Line 110 is shown containing a call to task procedure P1( ). Line 115 is shown containing a call to task procedure P2( ) and the status returned by execution of P2( ) is assigned to a variable Status."

AAPA teaches "*keeping track of a set of rollback procedures corresponding to said set of task procedures, each of said set of rollback procedures being determined based on a combination corresponding to an executed task procedure contained in said set of task procedures, said combination being contained in said plurality of combinations specified in said user program*," see Fig. 1, par. 7, "Thus, in one prior approach, a programmer may have to design programs to keep track of the specific tasks that have completed, and rollback the completed tasks if an atomic transaction is to be aborted," and par. 25, "Control passes to line 125 if an error has occurred, to line 140 otherwise. Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )," where the claimed "combination" is, for example, the referenced Account( ), P1( ), and do-reverse-of-P1( ) combination. AAPA does not teach "*wherein said set of rollback procedures are kept track of external to said user program in response to said executing of the corresponding task procedures*." Lordi does, however, see Fig. 2 and col. 5, ll. 50-62, "A Perform routine 100 for an operation takes the same parameters as does the corresponding native routine and generally executes the following steps… makes a log entry by creating the entry and appending it to a transaction log (a log database), the entry containing information needed to commit and roll back, including the information required to call the Finalize and Undo routines," where the claimed "task procedure" is the referenced "Perform routine" and the claimed "rollback procedure" is the referenced "Undo routine." The referenced log "keeps track" of the

Undo routines and is external to the user program, as it is a "log database." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Lordi's teachings would have allowed AAPA's method to gain permanent, non-volatile storage of the information and procedures required to rollback the system to a previous state without relying on the possibly crashed user program, as in AAPA, see Lordi col. 6, ll. 3-11.

AAPA teaches "*and executing said set of rollback procedures in a reverse order of said sequential order if said atomic transaction is to be aborted*," see Fig. 1 and par. 25, "Control passes to line 125 if an error has occurred, to line 140 otherwise. Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )."

AAPA teaches "*wherein said rollback procedure is specified as a separate procedure from said task procedure in said user program*," see Fig. 1 and par. 25, "Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )."

AAPA teaches "*wherein said user program contains groups of instructions to implement respective program logic for each of said task procedure and said rollback procedure*," see Fig. 1 and par. 23, "For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures. However, typical atomic transactions contain many task procedures," where the claimed "groups of instructions" are contained in the referenced "procedures," see Applicant's specification par. 35, which defines a procedure as "a group of instructions identified by a name."

AAPA teaches "*and whereby each user program has corresponding custom logic specified by a user for each of the rollback procedure*," see Fig. 1 and par. 23, "FIG. 1 contains pseudo-code illustrating the manner in which an example atomic transaction is implemented in a prior approach." Since a user writes the code, that user could add custom logic to the procedures.

AAPA does not teach "*requesting in a user program a transaction identifier for an atomic transaction*." Gostanian does, however, see Figs. 3, 5, col. 9, lines 1-21, "Each application client 302-308 is essentially an application program that preferably resides on a client computer 220 (FIG. 2)," col. 9, lines 27-42, "The application servers 332, 334 coordinate the requested database transactions for the application clients 302-308" and col. 13, line 61 – col. 14, line 9, "As with the 1PPC protocol 400 (FIG. 4), a manager process 516 of the coordinator 512 first assigns a unique transaction identification code 524 to the particular transaction," where the claimed "user program" is the referenced "application program." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.

AAPA does not teach "*generating said transaction identifier in a transaction manager in response to said requesting*." Gostanian does, however, see Fig. 5 and col. 13, line 61 – col. 14, line 9, "As with the 1PPC protocol 400 (FIG. 4), a manager process 516 of the coordinator 512 first assigns a unique transaction identification code 524 to the particular transaction." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have

allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.

2. AAPA does not teach "*The method of claim 1, wherein said transaction identifier is unique to each of the atomic transactions*." Gostanian does, however, see Fig. 5 and col. 13, line 61 – col. 14, line 9, "As with the 1PPC protocol 400 (FIG. 4), a manager process 516 of the coordinator 512 first assigns a unique transaction identification code 524 to the particular transaction." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.

5. AAPA teaches "*The method of claim 1, wherein said user program further comprises additional instruction for examining a status returned by execution of one of said task procedures and performing said aborting if said status indicates an error*," see Fig. 1 and par. 25, "In line 120, the variable Status is compared with ERROR1 (either a variable set ahead, or a constant value defined elsewhere) to determine whether an error has occurred in the execution of P2( )."

AAPA does not teach "*wherein said aborting is specified in said user program using an instruction containing said transaction identifier*." Lordi does, however, see col. 2, l. 66 - col. 3, l. 7, "To roll a transaction back (Step 46), the transaction master broadcasts a roll back message (including the identifier for the transaction) to all agents that participated in the transaction (Step 48). Each agent steps through its log (Step 50), and for each operation belonging to the transaction being rolled back, an undo routine may be invoked which will have the effect of

undoing the effects of the original operation (Step 52)." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Lordi's teachings would have allowed AAPA's method to gain permanent, non-volatile storage of the information and procedures required to rollback the system to a previous state without relying on the possibly crashed user program, as in AAPA, see Lordi col. 6, ll. 3-11.

6. AAPA teaches "*The method of claim 1, wherein said aborting is performed asynchronously*," see Fig. 1 and par. 28, "In line 160, the variable Status is compared with ERROR2 (which is similar to ERROR1, described above) to determine whether an error has occurred in the execution of P6( )."

7. AAPA teaches "*A computer readable storage medium carrying one or more sequences of instructions representing a user program for execution on a system, said user program implementing an atomic transaction, wherein execution of said one or more sequences of instructions by one or more processors contained in said system causes said system to perform the actions of*," see Fig. 1 and par. 23, "FIG. 1 contains pseudo-code illustrating the manner in which an example atomic transaction is implemented in a prior approach."

AAPA teaches "*specifying a plurality of combinations in said user program for execution in said system, wherein each of said plurality of combinations contains said variable, a task procedure, and a rollback procedure*," see Fig. 1, par. 23, "For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures… Account1( ) is shown containing program logic in lines 110 through 199," par. 24, "Line 110 is shown containing a call to task procedure P1( ). Line 115 is

shown containing a call to task procedure P2( ),” and par. 25, “Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( ),” where the claimed “combinations” are the referenced Account( ), P( ), and do-reverse-of-P( ) combinations, and where Account( ) is the atomic transaction identifier. According to Applicant’s specification at par. 69, “Even though the example above are shown specifying the combination in the form of a single line of code (procedure call), multiple lines can be used in alternative embodiments.” Thus, it is irrelevant that the referenced combinations are not contained in a single procedure call.

AAPA teaches “*wherein each combination indicates that the rollback procedure is to be executed if the execution of the corresponding task procedure in the combination is completed and if said atomic transaction is to be aborted*,” see Fig. 1 and pars. 25-26, “Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( ). The occurrence of an error generally represents an example situation in which the atomic transaction is to be aborted.”

AAPA teaches “*wherein said task procedure implements a part of said atomic transaction and said rollback procedure is designed to rollback said task procedure, wherein said variable in each of said plurality of combinations specifies said identifier generated by said transaction manager*,” see Fig. 1 and par. 23, “For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures.”

AAPA teaches “*executing a set of task procedures in a sequential order, wherein said set of task procedures are contained in said task procedures specified in said plurality of*

combinations," see Fig. 1 and par. 24, "Line 110 is shown containing a call to task procedure P1( ). Line 115 is shown containing a call to task procedure P2( ) and the status returned by execution of P2( ) is assigned to a variable Status."

AAPA teaches "*wherein said plurality of combinations and said abort procedure are contained in a said user program*," see Fig. 1 and par. 25, "Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )."

AAPA teaches "*wherein said user program contains groups of instructions to implement respective program logic for each of said task procedure and said rollback procedure*." AAPA does, however, see Fig. 1 and par. 23, "For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures. However, typical atomic transactions contain many task procedures," where the claimed "groups of instructions" are contained in the referenced "procedures," see Applicant's specification par. 35, which defines a procedure as "a group of instructions identified by a name."

AAPA teaches "*whereby each user program has corresponding custom logic specified by a user for each of the rollback procedure*," see Fig. 1 and par. 23, "FIG. 1 contains pseudo-code illustrating the manner in which an example atomic transaction is implemented in a prior approach." Since a programmer writes the code, that programmer could add custom logic to the procedures.

AAPA does not teach "*requesting an identifier in said user program from a transaction manager for said atomic transaction, said transaction manager being provided external to said*

user program, wherein said transaction manager generates a unique value as said identifier and provides said identifier to said user program." Gostanian does, however, see Figs. 3, 5, col. 9, lines 1-21, "Each application client 302-308 is essentially an application program that preferably resides on a client computer 220 (FIG. 2)," col. 9, lines 27-42, "The application servers 332, 334 coordinate the requested database transactions for the application clients 302-308" and col. 13, line 61 – col. 14, line 9, "As with the 1PPC protocol 400 (FIG. 4), a manager process 516 of the coordinator 512 first assigns a unique transaction identification code 524 to the particular transaction," where the claimed "user program" is the referenced "application program." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.

AAPA does not teach "*setting a variable to equal said identifier in said user program.*" Lordi does, however, see col. 6, l. 56 – col. 7, l. 5, "In a second way of deploying the routines, a runtime library is created containing the routines, plus routines to start, finalize, and undo transactions… Upon abnormal termination, the application invokes the roll back process, which in turn scans the log and invokes the appropriate Undo routines," where the claimed "user program" is the claimed "routines" and it would have been obvious to "set a variable to equal said identifier" so that the Perform and Undo routines would know which transactions to perform or undo. Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Lordi's teachings would have allowed AAPA's method to gain permanent, non-volatile storage of the information

and procedures required to rollback the system to a previous state without relying on the possibly crashed user program, as in AAPA, see Lordi col. 6, ll. 3-11.

AAPA does not teach "*and aborting said atomic transaction by specifying, in said user program, said identifier associated with an abort procedure to cause a set of rollback procedures to be executed in a reverse order of said sequential order*." Lordi does, however, see col. 2, l. 66 - col. 3, l. 7, "To roll a transaction back (Step 46), the transaction master broadcasts a roll back message (including the identifier for the transaction) to all agents that participated in the transaction (Step 48). Each agent steps through its log (Step 50), and for each operation belonging to the transaction being rolled back, an undo routine may be invoked which will have the effect of undoing the effects of the original operation (Step 52)." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Lordi's teachings would have allowed AAPA's method to gain permanent, non-volatile storage of the information and procedures required to rollback the system to a previous state without relying on the possibly crashed user program, as in AAPA, see Lordi col. 6, ll. 3-11. AAPA does teach "*each of said set of rollback procedures being determined based on a combination corresponding to an executed task procedure contained in said set of task procedures*," see Fig. 1 and par. 25, "Control passes to line 125 if an error has occurred, to line 140 otherwise. Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )."

8. AAPA teaches "*The computer readable storage medium of claim 7, wherein said specifying comprises including each of said plurality of combinations in a single procedure call*," see Fig. 1 and par. 23, "For ease of understanding, atomic transaction Account1( ) (starting

at line 105) is shown containing only few task procedures and desired roll-back procedures… Account1( ) is shown containing program logic in lines 110 through 199," where the claimed "single procedure call" is the referenced "Account1( )."

9. AAPA teaches "*The computer readable storage medium of claim 7, further comprising examining a status returned by execution of one of said task procedures and performing said aborting if said status indicates an error*," see Fig. 1 and par. 25, "In line 120, the variable Status is compared with ERROR1 (either a variable set ahead, or a constant value defined elsewhere) to determine whether an error has occurred in the execution of P2( )."

10. AAPA teaches "*A computer readable storage medium carrying one or more sequences of instructions for supporting implementation of a transaction manager which support an atomic transaction in a user program executing in a system, wherein execution of said one or more sequences of instructions by one or more processors contained in said system causes said system to perform the actions of*," see Fig. 1 and par. 23, "FIG. 1 contains pseudo-code illustrating the manner in which an example atomic transaction is implemented in a prior approach."

AAPA teaches "*receiving a plurality of combinations for execution from said user program, wherein each of said plurality of combinations contains said transaction identifier, a task procedure, and a rollback procedure*," see Fig. 1, par. 23, "For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures… Account1( ) is shown containing program logic in lines 110 through 199," par. 24, "Line 110 is shown containing a call to task procedure P1( ). Line 115 is shown containing a call to task procedure P2( )," and par. 25, "Lines 125 (do-

reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( ),” where the claimed “combinations” are the referenced Account( ), P( ), and do-reverse-of-P( ) combinations, and where Account( ) is the atomic transaction identifier. According to Applicant’s specification at par. 69, “Even though the example above are shown specifying the combination in the form of a single line of code (procedure call), multiple lines can be used in alternative embodiments.” Thus, it is irrelevant that the referenced combinations are not contained in a single procedure call.

AAPA teaches “*wherein each combination indicates that the rollback procedure is to be executed if the execution of the corresponding task procedure in the combination is completed and if said atomic transaction is to be aborted*,” see Fig. 1 and pars. 25-26, “Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( ). The occurrence of an error generally represents an example situation in which the atomic transaction is to be aborted.”

AAPA teaches “*wherein said task procedure implements a part of said atomic transaction and said rollback procedure is designed to rollback said task procedure*,” see Fig. 1, par. 23, “For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures.”

AAPA teaches “*executing a set of task procedures in a sequential order according to said user program, wherein said set of task procedures are contained in said task procedures specified in said plurality of combinations*,” see Fig. 1 and par. 24, “Line 110 is shown containing a call to task procedure P1( ). Line 115 is shown containing a call to task procedure P2( ) and the status returned by execution of P2( ) is assigned to a variable Status.”

AAPA teaches "*keeping track of a set of rollback procedures corresponding to said set of task procedures, each of said set of rollback procedures being determined based on a combination corresponding to an executed task procedure contained in said set of task procedures, said combination being contained in said plurality of combinations specified in said user program*," see Fig. 1 and par. 25, "Control passes to line 125 if an error has occurred, to line 140 otherwise. Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )," where the claimed "combination" is, for example, the referenced Account( ), P1( ), and do-reverse-of-P1( ) combination, and where Account( ) is the atomic transaction identifier.

AAPA teaches "*and executing said set of rollback procedures in a reverse order of said sequential order in response to receiving an abort request*," see Fig. 1 and par. 24, "Control passes to line 125 if an error has occurred, to line 140 otherwise. Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )." AAPA does not teach "*said abort request being received from said user program and containing said identifier*." Lordi does, however, see col. 2, l. 66 - col. 3, l. 7, "To roll a transaction back (Step 46), the transaction master broadcasts a roll back message (including the identifier for the transaction) to all agents that participated in the transaction (Step 48). Each agent steps through its log (Step 50), and for each operation belonging to the transaction being rolled back, an undo routine may be invoked which will have the effect of undoing the effects of the original operation (Step 52)." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Lordi's teachings would have allowed AAPA's method to gain permanent, non-volatile

storage of the information and procedures required to rollback the system to a previous state without relying on the possibly crashed user program, as in AAPA, see Lordi col. 6, ll. 3-11.

AAPA teaches "*wherein said rollback procedure is specified as a separate procedure from said task procedure in said user program*," see Fig. 1 and par. 25, "Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )."

AAPA teaches "*wherein said user program contains groups of instructions to implement respective program logic for each of said task procedure and said rollback procedure*," see Fig. 1 and par. 23, "For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures. However, typical atomic transactions contain many task procedures," where the claimed "groups of instructions" are contained in the referenced "procedures," see Applicant's specification par. 35, which defines a procedure as "a group of instructions identified by a name."

AAPA teaches "*whereby each user program has corresponding custom logic specified by a user for each of the rollback procedures*," see Fig. 1 and par. 23, "FIG. 1 contains pseudo-code illustrating the manner in which an example atomic transaction is implemented in a prior approach." Since a programmer writes the code, that programmer could add custom logic to the procedures.

AAPA does not teach "*generating an identifier for said atomic transaction for said user program*." Gostanian does, however, see Fig. 5 and col. 13, line 61 – col. 14, line 9, "As with the 1PPC protocol 400 (FIG. 4), a manager process 516 of the coordinator 512 first assigns a unique transaction identification code 524 to the particular transaction." Thus, it would have

been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.

AAPA does not teach "*providing said identifier to said user program*." Gostanian does, however, see Fig. 5 and col. 13, l. 61 – col. 14, l. 9, "As with the 1PPC protocol 400 (FIG. 4), a manager process 516 of the coordinator 512 first assigns a unique transaction identification code 524 to the particular transaction. Next, as shown by block 526, the manager process 516 forwards the transaction to each cohort 514 using an atomic multicast message 528." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.

AAPA does not explicitly teach "*wherein said transaction manager is provided external to user programs including said user program*." Gostanian does, however, see Fig. 5, where the claimed "transaction manager" is the referenced "coordinator" and the claimed "user program" resides on the referenced "cohort" since AAPA's user program is comparable to Gostanian's cohorts, since it executes the various procedures of the atomic transaction. Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.

13. AAPA does not teach "*The computer readable storage medium of claim 10, wherein said transaction identifier is generated to be unique for each atomic transaction*." Gostanian

does, however, see Fig. 5 and col. 13, line 61 – col. 14, line 9, "As with the 1PPC protocol 400 (FIG. 4), a manager process 516 of the coordinator 512 first assigns a unique transaction identification code 524 to the particular transaction." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.

16. AAPA teaches "*a computer readable medium to store and provide said plurality of instructions to said memory, wherein execution of said plurality of instructions by said processing unit causes said computer system to support implementation of atomic transactions in a programming environment by performing the operations of*," see Fig. 1 and par. 23, "FIG. 1 contains pseudo-code illustrating the manner in which an example atomic transaction is implemented in a prior approach."

AAPA teaches "*specify in said user program a plurality of combinations for execution in a sequential order, wherein each of said plurality of combinations contains said transaction identifier, a task procedure, and a rollback procedure*," see Fig. 1, par. 23, "For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures… Account1( ) is shown containing program logic in lines 110 through 199," par. 24, "Line 110 is shown containing a call to task procedure P1( ). Line 115 is shown containing a call to task procedure P2( )," and par. 25, "Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )," where the claimed "combinations" are the referenced Account( ), P( ), and do-reverse-of-P( ) combinations, and where Account( ) is the atomic

transaction identifier. According to Applicant's specification at par. 69, "Even though the example above are shown specifying the combination in the form of a single line of code (procedure call), multiple lines can be used in alternative embodiments." Thus, it is irrelevant that the referenced combinations are not contained in a single procedure call.

AAPA teaches "*wherein each combination indicates that the rollback procedure is to be executed if the execution of the corresponding task procedure in the combination is completed and if said atomic transaction is to be aborted*," see Fig. 1 and pars. 25-26, "Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( ). The occurrence of an error generally represents an example situation in which the atomic transaction is to be aborted."

AAPA teaches "*wherein said task procedure implements a part of said atomic transaction and said rollback procedure is designed to rollback said task procedure, wherein said rollback procedure is specified as a separate procedure from said task procedure*," see Fig. 1, par. 23, "For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures."

AAPA teaches "*execute a set of task procedures in a sequential order according to said user program, wherein said set of task procedures are contained in said task procedures specified in said plurality of combinations*," see Fig. 1 and par. 24, "Line 110 is shown containing a call to task procedure P1( ). Line 115 is shown containing a call to task procedure P2( ) and the status returned by execution of P2( ) is assigned to a variable Status."

AAPA teaches "*keep track of a set of rollback procedures corresponding to said set of task procedures, each of said set of rollback procedures being determined based on a*

combination corresponding to an executed task procedure contained in said set of task procedures, said combination being contained in said plurality of combinations specified in said user program," see Fig. 1 and par. 25, "Control passes to line 125 if an error has occurred, to line 140 otherwise. Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )," where the claimed "combination" is, for example, the referenced Account( ), P1( ), and do-reverse-of-P1( ) combination, and where Account( ) is the atomic transaction identifier. AAPA does not teach "*wherein said set of rollback procedures are kept track of external to said user program in response to said executing of the corresponding task procedures*." Lordi does, however, see Fig. 2 and col. 5, ll. 50-62, "A Perform routine 100 for an operation takes the same parameters as does the corresponding native routine and generally executes the following steps… makes a log entry by creating the entry and appending it to a transaction log (a log database), the entry containing information needed to commit and roll back, including the information required to call the Finalize and Undo routines," where the claimed "task procedure" is the referenced "Perform routine" and the claimed "rollback procedure" is the referenced "Undo routine." The referenced log "keeps track" of the Undo routines and is external to the user program, as it is a "log database." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Lordi's teachings would have allowed AAPA's method to gain permanent, non-volatile storage of the information and procedures required to rollback the system to a previous state without relying on the possibly crashed user program, as in AAPA, see Lordi col. 6, ll. 3-11.

AAPA teaches "*and execute said set of rollback procedures in a reverse order of said sequential order if said atomic transaction is to be aborted, wherein said rollback procedures are identified according to said keeping*," see Fig. 1 and par. 24, "Control passes to line 125 if an error has occurred, to line 140 otherwise. Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )."

AAPA teaches "*wherein said user program contains groups of instructions to implement respective program logic for each of said task procedure and said rollback procedure*," see Fig. 1 and par. 23, "For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures. However, typical atomic transactions contain many task procedures," where the claimed "groups of instructions" are contained in the referenced "procedures," see Applicant's specification par. 35, which defines a procedure as "a group of instructions identified by a name."

AAPA teaches "*whereby each user program has corresponding custom logic specified by a user for each of the rollback procedures*," see Fig. 1 and par. 23, "FIG. 1 contains pseudo-code illustrating the manner in which an example atomic transaction is implemented in a prior approach." Since a programmer writes the code, that programmer could add custom logic to the procedures.

AAPA does not teach "*A computer system comprising*." Gostanian does, however, see Fig. 2 and col. 7, lines 46-62, "FIG. 2 is a block diagram of a database system 200." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have

allowed AAPA's method to gain a common means of implementing database operations, see Gostanian col. 7, ll. 46-62.

AAPA does not teach "*a memory storing a plurality of instructions*," see Fig. 2 and col. 8, lines 14-24, "As shown in FIG. 2, a typical hardware configuration of a client 220 includes a central processing unit (CPU) 222 coupled between a memory 224." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of implementing database operations, see Gostanian col. 7, ll. 46-62.

AAPA does not teach "*and a processing unit coupled to said memory and executing said plurality of instructions*." Gostanian does, however, see Fig. 2, e.g. "CPU" 222. Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of implementing database operations, see Gostanian col. 7, ll. 46-62.

AAPA does not teach "*request in a user program, a transaction identifier for an atomic transaction*." Gostanian does, however, see Figs. 3, 5, col. 9, lines 1-21, "Each application client 302-308 is essentially an application program that preferably resides on a client computer 220 (FIG. 2)," col. 9, lines 27-42, "The application servers 332, 334 coordinate the requested database transactions for the application clients 302-308" and col. 13, line 61 – col. 14, line 9, "As with the 1PPC protocol 400 (FIG. 4), a manager process 516 of the coordinator 512 first assigns a unique transaction identification code 524 to the particular transaction," where the

claimed "user program" is the referenced "application program." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.

AAPA does not teach "*generate said transaction identifier in a transaction manager in response to said requesting, wherein said transaction manager is provided external to said user program.*" Gostanian does, however, see Fig. 5 and col. 13, line 61 – col. 14, line 9, "As with the 1PPC protocol 400 (FIG. 4), a manager process 516 of the coordinator 512 first assigns a unique transaction identification code 524 to the particular transaction." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.

17. AAPA does not teach "*The computer system of claim 16, wherein said transaction identifier is unique to each of the atomic transactions.*" Gostanian does, however, see Fig. 5 and col. 13, line 61 – col. 14, line 9, "As with the 1PPC protocol 400 (FIG. 4), a manager process 516 of the coordinator 512 first assigns a unique transaction identification code 524 to the particular transaction." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.

20. AAPA teaches "*The computer system of claim 16, wherein the actions performed by said computer system further comprise examine a status returned by execution of one of said task procedures and to perform said aborting if said status indicates an error*," see Fig. 1.

21. Gostanian teaches "*The computer system of claim 16, wherein the actions performed by said computer system further comprise execute said rollback procedures asynchronously*," see Fig. 1.

25. AAPA teaches "*The computer readable storage medium of claim 7, wherein said rollback procedure is specified as a separate procedure from said task procedure in said user program*," see Fig. 1.

Claims 3-4, 14-15 and 18-19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Applicant's Admitted Prior Art, Fig. 1 and Specification pars. 3-7 and 22-33 ("AAPA"), in view of Gostanian et al., U.S. 5,781,910 ("Gostanian"), in view of Lordi et al., U.S. 5,857,204 ("Lordi"), and in view of Raz, U.S. 5,701,480 ("Raz").

3. AAPA does not teach "*The method of claim 1, wherein said keeping comprises storing data representing said rollback procedures in a stack*." Raz does, however, see col. 19, lines 51-59, "the transaction scheduler responds to an interrupt by removing the context of the interrupted transaction from the processor stack of the digital computer… The context includes the value of the program counter which points to the interrupted memory location in the transaction program." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Raz's teachings

would have allowed AAPA's method to gain way to keep track of transactions, see Raz col. 19, lines 51-59.

4. AAPA does not teach "*The method of claim 3, wherein said stack is stored in a memory*." Raz does, however, see col. 2, lines 7-24, "the operating system typically provides an established set of memory management procedures that can be invoked or called from an application program to define a 'recovery unit,'" where the "stack" in the reference is part of the "recovery unit." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Raz's teachings would have allowed AAPA's method to gain way to keep track of transactions, see Raz col. 19, lines 51-59.

14. AAPA does not teach "*The computer readable storage medium of claim 10, wherein said set of rollback procedures are represented in the form of a stack*." Raz does, however, see col. 19, lines 51-59, "the transaction scheduler responds to an interrupt by removing the context of the interrupted transaction from the processor stack of the digital computer… The context includes the value of the program counter which points to the interrupted memory location in the transaction program." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Raz's teachings would have allowed AAPA's method to gain way to keep track of transactions, see Raz col. 19, lines 51-59.

15. AAPA does not teach "T*he computer readable storage medium of claim 14, wherein said stack is stored in a memory*." Raz does, however, see col. 2, lines 7-24, "the operating system typically provides an established set of memory management procedures that can be

invoked or called from an application program to define a 'recovery unit'", where the "stack" in the reference is part of the "recovery unit." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Raz's teachings would have allowed AAPA's method to gain way to keep track of transactions, see Raz col. 19, lines 51-59.

18. AAPA does not teach "*The computer system of claim 16, wherein the actions performed by said computer system further comprise store data representing said rollback procedures in a stack to perform said keep.*" Raz does, however, see col. 19, lines 51-59, "the transaction scheduler responds to an interrupt by removing the context of the interrupted transaction from the processor stack of the digital computer… The context includes the value of the program counter which points to the interrupted memory location in the transaction program." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Raz's teachings would have allowed AAPA's method to gain way to keep track of transactions, see Raz col. 19, lines 51-59.

19. AAPA do not teach "*The computer system of claim 18, wherein said stack is stored in a memory.*" Raz does, however, see col. 2, lines 7-24, "the operating system typically provides an established set of memory management procedures that can be invoked or called from an application program to define a 'recovery unit,'" where the "stack" in the reference is part of the "recovery unit." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Raz's

teachings would have allowed AAPA's method to gain way to keep track of transactions, see Raz col. 19, lines 51-59.

(10) Response to Argument

Applicant's Argument VII.A.1

As per applicant's argument that the combination of references does not teach requesting a transaction identifier, generating said transaction identifier, and specifying combinations in the user program that contain said transaction identifier as recited in independent claims 1, 7, and 10, the examiner respectfully disagrees. For convenience, the examiner has reproduced the rejection of claim 7:

> AAPA does not teach "*requesting an identifier in said user program from a transaction manager for said atomic transaction, said transaction manager being provided external to said user program, wherein said transaction manager generates a unique value as said identifier and provides said identifier to said user program.*" Gostanian does, however, see Figs. 3, 5, col. 9, lines 1-21, "Each application client 302-308 is essentially an application program that preferably resides on a client computer 220 (FIG. 2)," col. 9, lines 27-42, "The application servers 332, 334 coordinate the requested database transactions for the application clients 302-308" and col. 13, line 61 – col. 14, line 9, "As with the 1PPC protocol 400 (FIG. 4), a manager process 516 of the coordinator 512 first assigns a unique transaction identification code 524 to the particular transaction," where the claimed "user program" is the referenced "application program." Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to combine the teachings of the cited references because Gostanian's teachings would have allowed AAPA's method to gain a common means of identifying transactions, see Lordi col. 13, ll. 48-62.
>
> AAPA does not teach "*setting a variable to equal said identifier in said user program.*" Lordi does, however, see col. 6, l. 56 – col. 7, l. 5, "In a second way of deploying the routines, a runtime library is created containing the routines, plus routines to start, finalize, and undo transactions… Upon abnormal termination, the application invokes the roll back process, which in turn scans the log and invokes the appropriate Undo routines," where the claimed "user program" is the claimed "routines" and it would have been obvious to "set a variable to equal said identifier" so that the Perform and Undo routines would know which transactions to perform or undo. Thus, it would have been obvious to one of ordinary skill in the database art at the time of the invention to

> combine the teachings of the cited references because Lordi's teachings would have allowed AAPA's method to gain permanent, non-volatile storage of the information and procedures required to rollback the system to a previous state without relying on the possibly crashed user program, as in AAPA, see Lordi col. 6, ll. 3-11.
>
> AAPA teaches "*specifying a plurality of combinations in said user program for execution in said system, wherein each of said plurality of combinations contains said variable, a task procedure, and a rollback procedure*," see Fig. 1, par. 23, "For ease of understanding, atomic transaction Account1( ) (starting at line 105) is shown containing only few task procedures and desired roll-back procedures… Account1( ) is shown containing program logic in lines 110 through 199," par. 24, "Line 110 is shown containing a call to task procedure P1( ). Line 115 is shown containing a call to task procedure P2( )," and par. 25, "Lines 125 (do-reverse-of-P2( )) and 130 (do-reverse-of-P1( )) respectively represent roll-back procedures corresponding to P2( ) and P1( )," where the claimed "combinations" are the referenced Account( ), P( ), and do-reverse-of-P( ) combinations and the claimed "transaction identifier" is the referenced "Account( )," since it is the name (i.e. "identifier") of the transaction..

AAPA teaches the claimed "combinations" (e.g. the combination of "Account()," "P1()," and "do-reverse-of-P1()"), but does not teach that "Account()" is a transaction identifier requested by the user program, generated by a transaction manager, and returned to the user program. Applicant does not argue that Gostanian and Lordi do not teach the claimed "requesting" and "generating," only that the references cannot be combined with AAPA to teach returning the transaction identifier to the user program. (Claim 1, however, does not recite this limitation, so the argument is moot with respect to that claim.)

The examiner respectfully disagrees with applicant's argument. When combined with Gostanian's programs and Lordi's routines, both "user programs," it would be obvious to return the transaction identifier to AAPA's user program. Specifically, Lordi teaches that upon abnormal termination, the application invokes the rollback process, which in turn scans the log and invokes the appropriate undo routines (col. 6, l. 56 – col. 7, l. 5). In order for the rollback process to know what transactions to undo, it would have to be provided with a transaction identifier. Thus, when combined with AAPA, which keeps track of rollback procedures within

the user program, it would be obvious to provide the transaction identifier to the user program, so that the user program could manage which transactions to roll back.

In such a case, even if "Account()" were not an assigned transaction identifier (as the name of the transaction it "identifies" the transaction), one would have to be requested and returned to the user program in view of Lordi. Thus, the combination of references teaches the claimed "requesting," "generating," and "providing" limitations.

As per applicant's argument that "if" does not denote an optional limitation, the examiner respectfully disagrees. The claims recite that each combination indicates that the rollback procedure is to be executed only "if" two conditions are met. If both conditions are not met, nothing happens. The claims do not recite "determining" whether or not the conditions exist, which would be a required limitation. Thus, the limitation is optional, since it need never occur.

Even if the limitation included a mandatory determination of whether the two conditions existed, the references teach the limitation. Applicant argues that there is "no inherent logic to the structure of Figure 1, which makes the tuple akin to the claimed combination." The claims, however, do not require any such "logic." Even if they did, there is a clear link between the name of a transaction and the procedure and rollback procedure calls it contains. Further, the specification explicitly states that the combinations need not be a single procedure call (see par. 69, "Even though the example [*sic*] above are shown specifying the combination in the form of a single line of code (procedure call), multiple lines can be used in alternative embodiments"). Thus, it is irrelevant that the referenced "combinations" are not contained in a single procedure call.

As per applicant's argument that the claimed "keep track of a set of rollback procedures" (claim 16) cannot be equated with the programmer's mental effort, the examiner respectfully disagrees. However, that is not the examiner's position. Rather, once the programmer writes the program, the machine executing the program must somehow "keep track" of the various procedure and rollback procedure calls being executed in its memory. Thus, AAPA teaches the claimed limitation.

Applicant's Argument VII.A.2

As per applicant's argument that the references cannot be combined, the examiner respectfully disagrees. When combined with Gostanian's programs and Lordi's routines, both "user programs," it would be obvious to return the transaction identifier to AAPA's user program. Specifically, Gostanian teaches requesting and generating transaction identifiers. The application program requests a transaction and the manager assigns a transaction identifier (Fig. 5, steps 510 and 524). That means that in requesting a transaction, the application program is also requesting that a transaction identifier be generated. Lordi teaches that upon abnormal termination, the application invokes the rollback process, which in turn scans the log and invokes the appropriate undo routines (col. 6, l. 56 – col. 7, l. 5). In order for the rollback process to know what transactions to undo, it would have to be provided with a transaction identifier. Thus, when combined with AAPA, which keeps track of rollback procedures within the user program, it would be obvious to provide the transaction identifier to the user program, so that the user program could manage which transactions to roll back.

Applicant's Argument VII.A.3

As per applicant's argument that at least one of the references must provide the alleged benefits of the claimed invention, the examiner respectfully disagrees. That is not the standard for obviousness. If it were, all rejections would be 102 rejections. Rather, MPEP 2143 provides several exemplary (and therefore non-exclusive) rationales that support a conclusion of obviousness. Here, several rationales, such as (A), (E), (F), and (G), could apply. Specifically, the modifications to AAPA (requesting and receiving a transaction identifier, since applicant has not claimed around AAPA's "combinations") proposed by the combination of references are known and can be predictably applied to AAPA. Thus, the rejection is proper and no reference must provide the alleged benefits of the claimed invention.

Applicant's Argument VII.A.4

The declaration under 37 CFR 1.132 filed 11 December 2009 is insufficient to overcome the rejection of claims 1-10, 13-21, and 25 based upon 35 U.S.C. 103 as set forth in the last Office action because the showing is not commensurate in scope with the claims. See MPEP § 716. Long-felt need requires a showing that (1) the need was a persistent one that was recognized by those of ordinary skill in the art, (2) the need was not satisfied by another before the invention by applicant, and (3) the invention satisfies the need. MPEP § 716.04.

The novelty of the invention appears to be that rollback procedures, defined in a user program and "combined" with a task procedure, are kept track of externally to the user program so that a transaction can be aborted by calling an abort procedure and only passing it the

transaction identifier. See Specification pars. 18-19. Proof of long felt need must show that there was a long felt need for the claimed invention. Representative claims 1 and 7, however, only recite part of the novelty of applicant's invention. Both claims recite that rollback procedures are combined with task procedures, but claim 1 only recites that rollback procedures are kept track of externally to the user program while claim 7 only recites passing the transaction identifier to an abort procedure to execute the rollback procedures. Thus, any proof of long felt need must show a long felt need for the distinct embodiments of claim 1 and 7.

Further, assuming that applicant's admitted prior art is the closest prior art to the invention (Declaration 31 August 2009, Point 10), and that the invention satisfies the need for custom rollback procedures within a user program (Declaration 31 August 2009, Point 15), applicant has still not satisfied the first prong of the three-part long felt need test. Although the Declaration filed 11 December 2009 objectively shows a long felt need to implement atomic transactions (Point 13), it does not show a long felt need for the custom rollback procedures recited in representative claims 1 and 7.

Thus, when all of the evidence is considered, the totality of the rebuttal evidence of nonobviousness fails to outweigh the evidence of obviousness. Applicant should not that long felt need is a secondary consideration, and, even if proven, will not necessarily overcome a strong 35 U.S.C. 103 rejection. Here, applicant's admitted prior art teaches most limitations of the claims. Lordi teaches the novel aspects of the claims, specifically keeping track of rollback procedures external to the user program and performing an abort procedure by specifying the transaction identifier. It would have been obvious to combine these references, as shown above. Applicant is therefore encouraged to amend the claims to overcome the prior art, or submit

arguments based on the references themselves as to why the combination of references is improper.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

Conclusion

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/Aaron Sanders/
Examiner, Art Unit 2168
3 December 2010

Conferees:

/Tim T. Vo/

Supervisory Patent Examiner, Art Unit 2168

/Pierre M. Vital/
Supervisory Patent Examiner, Art Unit 2156